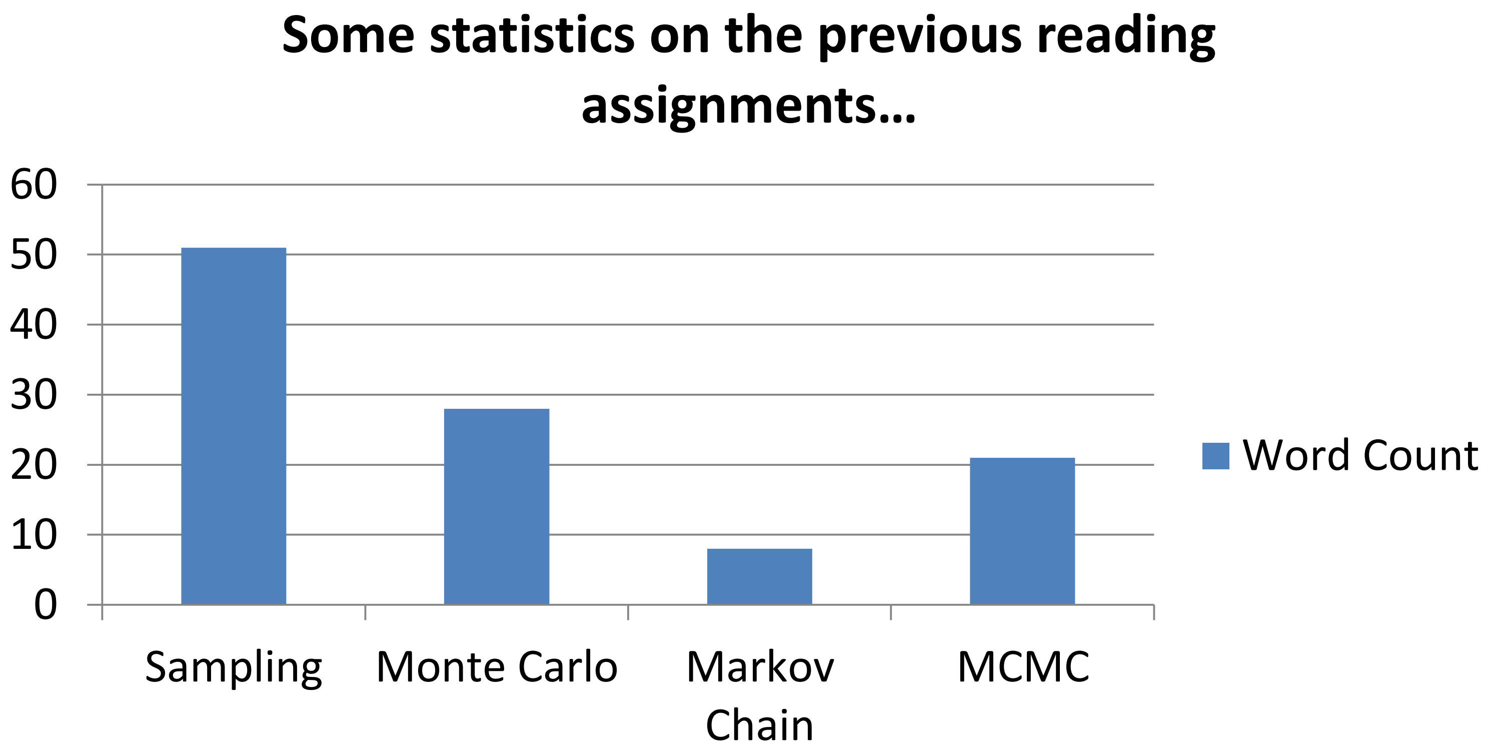
Sampling: Latin Hypercube & Markov Chain Monte Carlo

Matthias Blaschke

University of Augsburg

Memorial University of Newfoundland

Motivation



→We should learn something about these topics

Talk Outline

- Sampling Problem
- Monte Carlo Idea
- Markov Chains
- Markov Chain Monte Carlo Methods
- Latin Hypercube Sampling
- Comparison oft the presented Methods

The Sampling Problem

- D : Distribution over finite set X
- Given: Black-box access to the probability distribution function $p(x)$
- Goal: Output a sample of elements drawn according to $p(x)$

What can we do with the samples ?

- Analyze intractable posterior distribution
  - Remember the Roughier Paper ?

$$\Pr(y_f|z=\tilde{z}) = \int \Pr(y_f|x^*, z=\tilde{z}) \Pr(x^*|z=\tilde{z})\, dx^*$$

- Integration

$$I = \int_\theta g(\theta)p(\theta)\mathrm{d}\theta$$

$$I_{\mathrm{M}} = \frac{1}{\mathrm{M}} \sum_i^M g(\theta^{(i)})$$

Monte Carlo Methods

https://media.tag24.de/1/a/e/aecr2tfo7kgfv20w.jpg

- First experiments with Monte Carlo like methods by Enrico Fermi in the 1930s
- Modern version developed in the late 1940s by Stanislaw Ulam, while working on nuclear weapons
- Further work by John von Neuman

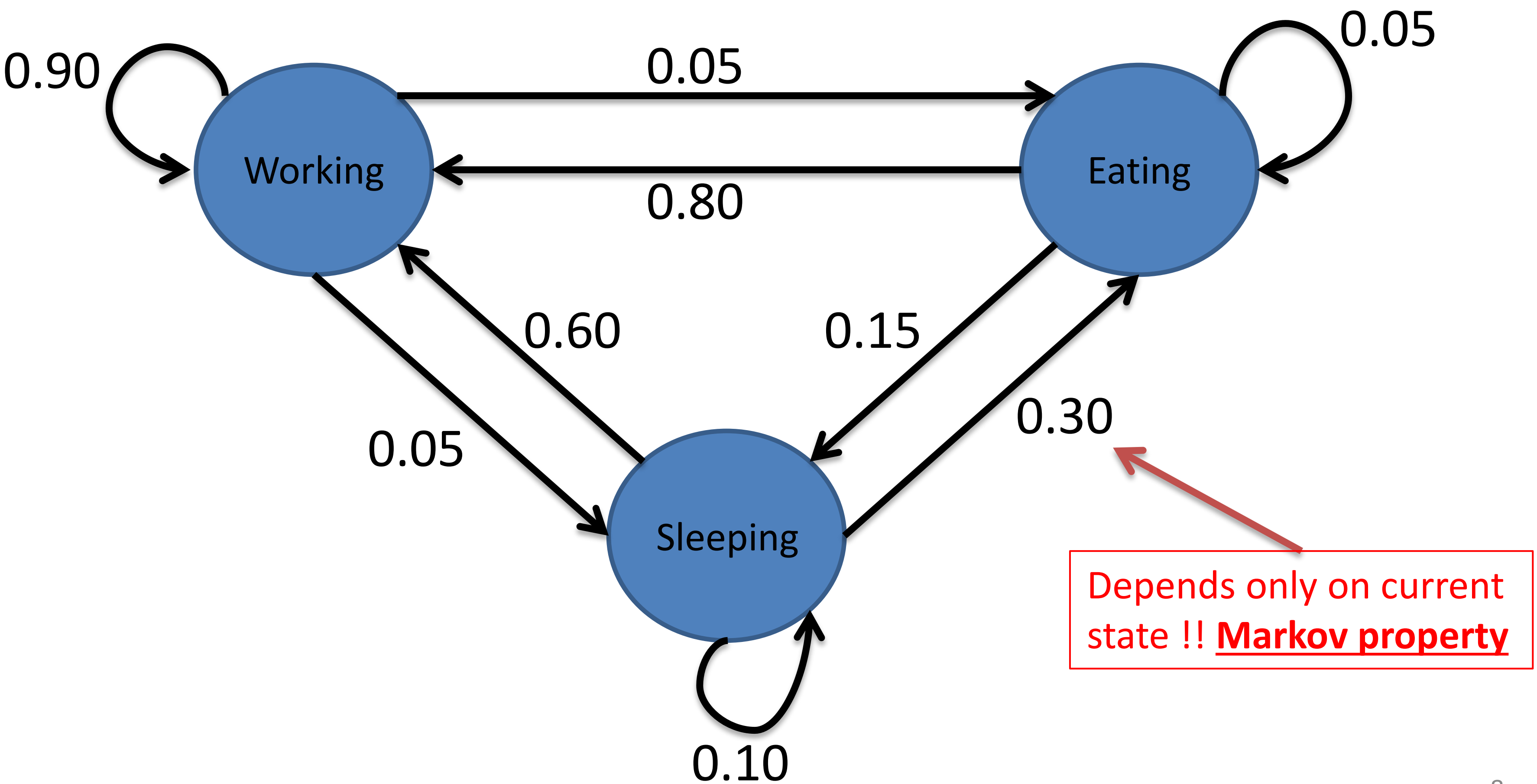
Monte Carlo Methods

General Monte Carlo pattern:

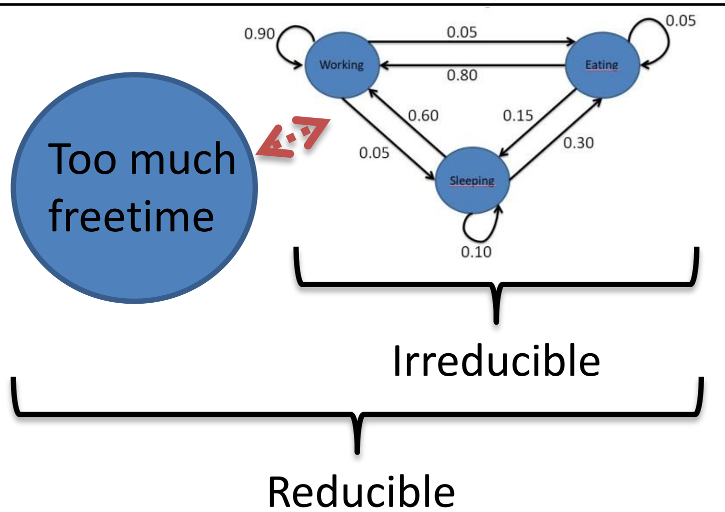
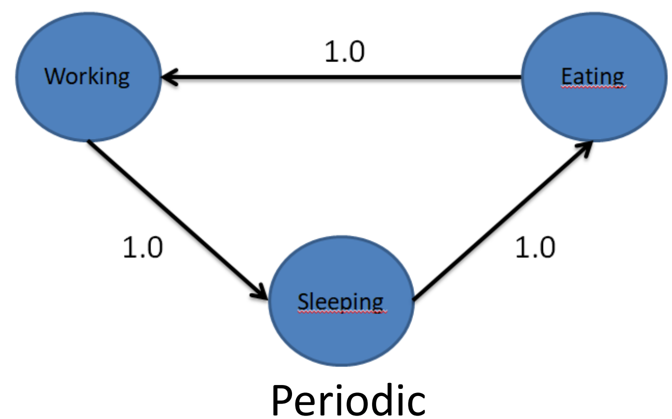
- Define a domain of possible inputs
- Generate inputs **randomly** from a probability distribution
- Perform a **deterministic computation** on the inputs
- Aggregate the results

Markov Chains

Markov state diagram of a student



Markov Chains

<table>
<tr>
<td>Irreducible</td>
<td>Possible to go from every state to every other state in one or more steps.</td>
<td>
</td>
</tr>
<tr>
<td>Periodic</td>
<td>One can return to a state only at regular intervals</td>
<td>
</td>
</tr>
</table>

Markov Chains

- **Irreducible** and **aperiodic** Markov Chains have **unique stationary distributions**!

  -> find this stationary distribution for the example using the Transition matrix

$$P(\text{Eating}|\text{Working})$$

$$T = \begin{pmatrix} 0.9 & 0.05 & 0.05 \\ 0.6 & 0.10 & 0.30 \\ 0.8 & 0.15 & 0.05 \end{pmatrix}$$

Markov Chains

-> State probabilities and Transition matrix

$$p = (p_{\text{work}}, p_{\text{eat}}, p_{\text{sleep}})$$

$$p^{j+1} = p^j T$$

-> Stationary distribution :

$$p = pT \quad \Leftrightarrow p^{\mathrm{T}} T^{\mathrm{T}} = p^{\mathrm{T}}$$

Find eigenvector with eigenvalue 1

$$p = (p_{\text{work}}, p_{\text{eat}}, p_{\text{sleep}}) \approx (0.88, 0.06, 0.06)$$

Markov Chain Monte Carlo Methods (MCMC)

- MCMC sampling sets up an **irreducible, aperiodic Markov Chain.**
- The stationary distribution equals posterior of interest **(for infinite chain lengths)**

  → Approximation

MCMC – Metropolis Hastings Algorithm

- Goal: Simulate $g(\theta|y)$ posterior
- Algorithm:

Markov property

1. begin with initial value θ^0
2. candidate value θ^* from proposal dens. $p(\theta^*|\theta^{t-1})$
3. Compute ratio R $$R = \frac{g(\theta^*)}{g(\theta^{t-1})}$$
4. Compute acceptance prob. $P = \min(R, 1)$
5. Accept θ^* with probability P

→ Sequence of θ will be distributed according to $g(\theta|y)$

MCMC – Metropolis Hastings Algorithm

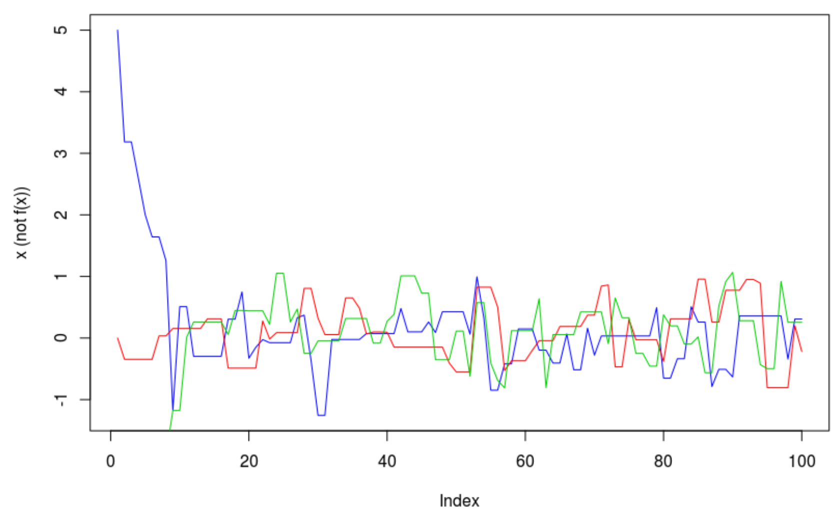
- -> Example (Sampling from unimodal and bimodal Gaussian)

MCMC Problems

- Values in chain must be **representative of the target distribution**
  - Explore full range without getting stuck
- Chain should be of **sufficient size**
  - Estimates accurate and stable
- For Metropolis-Hastings:
  - choose proposal distribution, Ratio, …
  - Rejection rate

MCMC Representativeness

- Methods to check convergence
  - Visual examination of chain trajectory (trace plot)
  - Burn-in period
  - Gelman-Rubin Test



MCMC Representativeness

- Methods to check accuracy
  - Calculate autocorrelation function (ACF)
  - Calculate effective sample size (ESS)

$$\mathrm{ESS} = N / \left(1 + 2 \sum_{k=1}^{\infty} \mathrm{ACF(k)} \right)$$

  - Calculate standard error of sample mean (MCSE)

$$\mathrm{MCSE} = \mathrm{SD} / \sqrt{\mathrm{ESS}}$$

MCMC Representativeness

- -> go back to example (Sampling from unimodal and bimodal Gaussian)

Metropolis Hastings Algorithm – Why it works

- Suppose we have a limited number of states θ_i
- Find Transition Matrix (→ Markov Chain example)

$$p(\theta - 2 \rightarrow \theta)$$

$$T = \begin{bmatrix} \ddots & p(\theta-2 \rightarrow \theta-1) & 0 & 0 & 0 \\ \ddots & p(\theta-1 \rightarrow \theta-1) & p(\theta-1 \rightarrow \theta) & 0 & 0 \\ 0 & p(\theta \rightarrow \theta-1) & p(\theta \rightarrow \theta) & p(\theta \rightarrow \theta+1) & 0 \\ 0 & 0 & p(\theta+1 \rightarrow \theta) & p(\theta+1 \rightarrow \theta+1) & \ddots \\ 0 & 0 & 0 & p(\theta+2 \rightarrow \theta+1) & \ddots \end{bmatrix}$$

$$p(\theta + 2 \rightarrow \theta)$$

Metropolis Hastings Algorithm – Why it works

- What are the elements in T ?

$$p(\theta - 1 \to \theta) = \text{propo}\mathbf{sal}(\theta) \times \text{acceptance}(\theta)$$

$$p(\theta - 1 \to \theta) = \text{sal}(\theta) \times \min\left(\frac{p(\theta - 1)}{p(\theta)}, 1\right)$$

$$p(\theta + i \to \theta) = \text{sal}(\theta) \times \min\left(\frac{p(\theta + i)}{p(\theta)}, 1\right)$$

$$p(\theta \to \theta) = 1 - \sum_{i \neq 0} p(\theta + i \to \theta)$$

- We know where we want to go...

$$\boxed{w = wT}$$ (distribution invariant under T)

$$w = 1/Z * (\cdots, p(\theta - 2), p(\theta - 1), p(\theta), p(\theta + 1), \cdots)$$

Metropolis Hastings Algorithm – Why it works

- Evaluate wT for the $p(\theta)$ element (min case-by-case)

  →always reduces to:

$$p(\theta) = p(\theta) * \underbrace{\sum_i \mathrm{sal}(\theta_\mathrm{i})}_{=1}$$

Comment on invariance

Always holds, if T satisfies detailed balance:

$$p(x)T(x \to x') = p(x')T(x' \to x)$$

But: detailed balance $\underset{\not\Leftrightarrow}{\Rightarrow}$ invariance $\Rightarrow$ stationary distribution

MCMC – Gibbs Sampling

- special case of the Metropolis–Hastings algorithm
- Parameter vector of interest $\theta = (\theta_1, \cdots, \theta_p)$
- **Given:** Set of conditional distributions

$$p(\theta_1 | \theta_2, \theta_3, \cdots, \theta_p, \mathrm{data})$$

$$p(\theta_2 | \theta_1, \theta_3, \cdots, \theta_p, \mathrm{data})$$

$$p(\theta_p | \theta_1, \cdots, \theta_{p-1}, \mathrm{data})$$

- **Goal:** Sample from $p(\theta | \mathrm{data})$

MCMC – Gibbs Sampling

- Algorithm
  1. Start with initial vector $\theta^0 = (\theta_1^0, \cdots, \theta_p^0)$
  2. Calculate new parameters θ_i (Gibbs Cycle)

$$\theta_1^t = p(\theta_1^t | \theta_2^{t-1}, \theta_3^{t-1}, \cdots, \theta_p^{t-1})$$

$$\theta_2^t = p(\theta_2^t | \theta_1^t, \theta_3^{t-1}, \cdots, \theta_p^{t-1})$$

$$\theta_p^t = p(\theta_p^t | \theta_1^t, \cdots, \theta_{p-1}^t)$$

MCMC – Gibbs Sampling

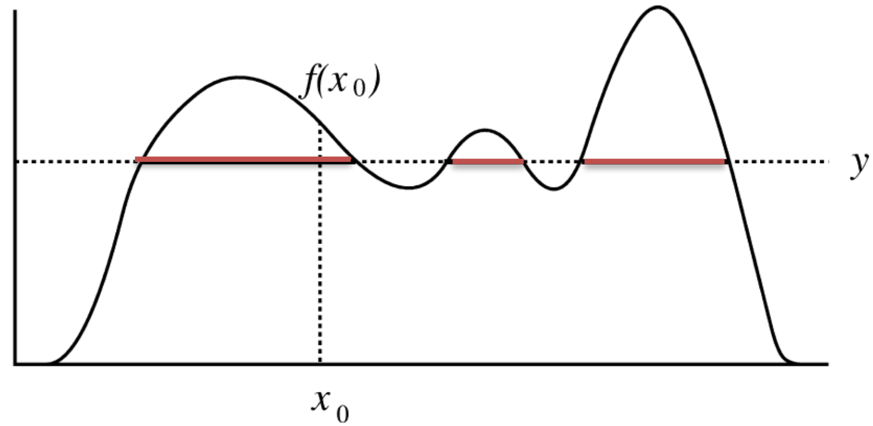
- -> Example (Sample from bivariate gaussian)

$$\begin{pmatrix} \theta_1 \\ \theta_2 \end{pmatrix} = N\left[\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}\right]$$

$$p(\theta_1|\theta_2, y) = N(\rho * \theta_2, 1 - \rho^2) = p\theta_2 + \sqrt{1 - p^2}N(0, 1)$$

$$p(\theta_2|\theta_1, y) = N(\rho * \theta_1, 1 - \rho^2) = p\theta_1 + \sqrt{1 - p^2}N(0, 1)$$

MCMC – Slice Sampling

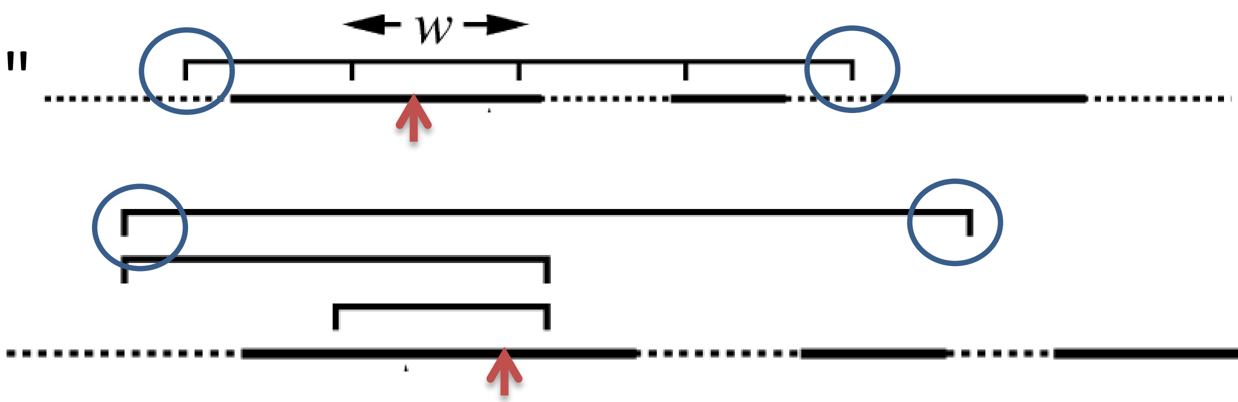


- Algorithm:
  1. Choose starting point x_0
  2. Draw a real value y uniformly from $(0, f(x_i))$
  3. Define horizontal "slice" $S = \{x : y < f(x)\}$
  4. Find an interval $I = (L, R)$ around x_i
  5. Draw the new point x_{i+1} from the **part of the slice** within this interval

MCMC – Slice Sampling

<u>Details to point 4: Find an interval around</u> x_i

- set I to the smallest interval that contains all of S → *not* feasible
- randomly pick an initial interval of size and expand it until both ends lie outside the slice
  - "Steeping out"
  - "Doubling"



MCMC – Slice Sampling

Details to point 5: Draw new point

Once Interval I has been found:

- Repeatedly sample uniformly from I until a point within S is found

  → **Speed up:** shrink I each time a point is drawn that is not in S

MCMC-Slice Sampling

Why Slice Sampling is a good Method compared to…

- Metropolis Hastings:
  - Less tuning parameters
  - Step sizes can be bigger
- Gibbs Sampling
  - No full conditional distributions for parameters needed

MCMC – Some improvements

- Run parallel chains
- Change parametrization of problem
- Thinning: use only every kth state to reduce autocorrelation
- …

Latin Hypercube Sampling (LHS)

- **Latin square** is an $n \times n$ array filled with n different symbols, each occurring exactly once in each row and exactly once in each column

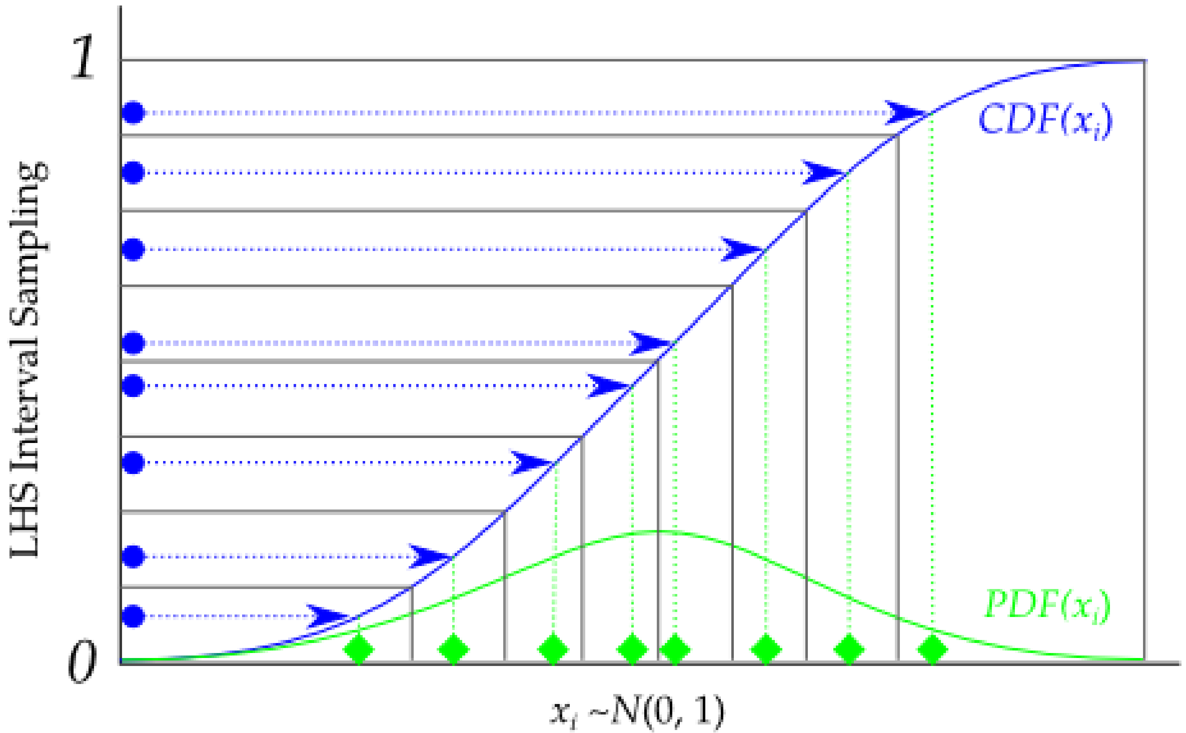
A	B	C
C	A	B
B	C	A

Latin Hypercube Sampling (LHS)

- How it works:
  - Represent each variable as is Cumulative Distribution Function (CDF)
  - Partition CDF into N regions (<-> Latin Hypercube)
  - Take sample from each region

→ Full range of the distribution is sampled

Latin Hypercube Sampling (LHS)



https://pythonhosted.org/pyDOE/_images/lhs_custom_distribution.png

Latin Hypercube Sampling (LHS)

- -> Example (Sampling from bivariate Gaussian)

Latin Hypercube vs. Markov Chain Monte Carlo

Why Latin Hypercube Sampling is a good Method…

- produces a clear depiction of each input distribution
- Avoid sampling artefacts

Literature

- Albert, Jim. *Bayesian computation with R*. Springer Science & Business Media, 2009.
- Hoff, Peter D. *A first course in Bayesian statistical methods*. Vol. 580. New York: Springer, 2009.
- Kruschke, John. *Doing Bayesian data analysis: A tutorial with R, JAGS, and Stan*. Academic Press, 2014.
- https://icme.hpc.msstate.edu/mediawiki/index.php/Latin_Hypercube_Sampling_(LHS) (last checked 2019-11-19)
- https://jeremykun.com/2015/04/06/markov-chain-monte-carlo-without-all-the-bullshit/ (last checked 2019-11-19)
- http://www.csri.utoronto.ca/pub/radford/slice-aos.pdf (last checked 2019-11-23)